| **Universidad Tecnológica Nacional**  **Facultad Regional Avellaneda** | | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos | | | | | | | | | | | | |
| Materia: Laboratorio de Programación II | | | | | | | | | | | | |
| Apellido: |  | | | | Fecha: | | | | 09/12/2021 | | | |
| Nombre: |  | | | | Docente(2): | | | |  | | | |
| División: | 2º | | | | Nota(2): | | | |  | | | |
| Legajo: |  | | | | Firma(2): | | | |  | | | |
| Instancia(1): | **PP** |  | **RPP** |  | | **SP** |  | **RSP** | |  | **FIN** | X |

**(1)** Las instancias validas son: 1er Parcial (**PP**), Recuperatorio 1er Parcial (**RPP**), 2do Parcial (**SP**), Recuperatorio 2do Parcial (**RSP**), Final (**FIN**). Marque con una cruz.

**(2)** Campos a ser completados por el docente.

**IMPORTANTE:**

* **2 (dos) errores en el mismo tema anulan su puntaje.**
* La correcta documentación y reglas de estilo de la cátedra serán evaluadas.
* El proyecto debe ser creado en .Net 5.
* Colocar sus datos personales en el nombre de la carpeta principal y la solución: Apellido.Nombre.Div. Ej: Pérez.Juan.2D. No sé corregirán proyectos que no sea identificable su autor.
* No se corregirán exámenes que no compilen.
* **Reutilizar** tanto código como crean necesario.
* Colocar nombre de la clase (en estáticos), **this** o **base** en todos los casos que corresponda.
* Aplicar los principios de los 4 pilares de la POO.

*TIEMPO MÁXIMO PARA RESOLVER EL EXAMEN* ***90*** *MINUTOS.*

Se realizará una aplicación para remover de series de nuestro backlog

**● Partir del formulario dado.**

**● Se deberá trabajar con .NET 5 y Visual Studio 2019.** Instrucciones para la descarga:

https://codeutnfra.github.io/programacion\_2\_laboratorio\_2\_apuntes/docs/introduccion/entorno-trabajo#instala ci%C3%B3n-de-net

● Correr el siguiente script:

CREATE DATABASE [20211209-FINAL];

GO

USE [20211209-FINAL];

GO

CREATE TABLE [20211209-FINAL].dbo.Visitas

(

Nombre VARCHAR(255) NOT NULL,

Fecha VARCHAR(255) NOT NULL,

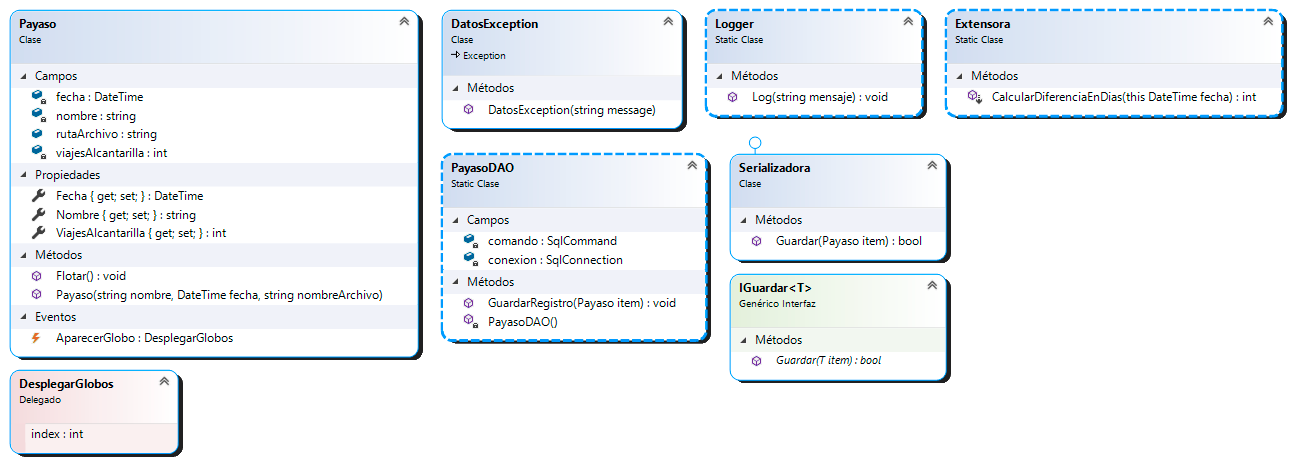
VisitasAlcantarilla INT NOT NULL,

Alumno VARCHAR(255) NOT NULL,

);

GO

1. Para comenzar, agregar a la solución un proyecto del tipo Biblioteca de Clases y reproducir el siguiente esquema:



**Extensora**

1. Generar un método de extensión para la clase **DateTime** que.
   * 1. Se va a encargar de restarle a la fecha actual la instancia de la fecha que vamos a usar retornando un objeto de tipo TimeSpan.
     2. El método va a encargar se de restarle a la fecha actual(DateTime.Now) la fecha que de la instancia que extendemos, esta operación va a retornar un objeto de tipo **TimeSpan**
     3. Utilizar la propiedad TotalDays de TimeSpan para obtener la cantidad de días resultante (castear a int).

**Payaso**

1. Todos los atributos de la clase van a ser privados.
2. Generar un constructor que reciba e inicialice todos los atributos.
3. El atributo *estático* rutaArchivo se va a generar concatenando el nombre del archivo que recibe el constructor por parámetro con la ruta por defecto que en este caso apunta hacia el escritorio.
4. El atributo viajesAlcantarilla se va a inicializar en el constructor a partir del método de extensión de la clase DateTime.
5. Tendrá tres propiedades que van a exponer a los atributos de la clase.
6. El delegado DesplegarGlobos va a recibir un entero como parámetro
7. Declarar el evento AparacerGlobo del tipo DesplegarGlobos
8. El método ***Flotar*** realizará lo siguiente:
   * 1. Suspenderá el hilo por 2 segundos.
     2. Usar una estructura repetitiva para lanzar el evento la cantidad de veces necesarias para que los globos sean visibles(5).
     3. Avisar mediante el evento Aparecer globo el índice del globo que debe aparecer(utilizar la estructura repetitiva que se habló en el ítem anterior),validar que el evento no sea nulo.
9. Una vez finalizada la pasada se deberá llamar al método estático de la clase **PayasoDao** para guardar el objeto payaso.

**DatosException**

1. Crear una excepción llamada **DatosException**. Con un constructor inicializando el mensaje de la excepción, esta excepción va a ser utilizada para el manejo de archivos.

**PayasoDAO**

1. Se deberá instalar el ***paquete NuGet System.Data.SqlClient.*** Si necesita ayuda pregunte al profesor a cargo.
2. Será la clase que represente la conexión con nuestra base de datos SQL.
3. Dicha clase será estática, y tendrá 2 atributos de tipo **SqlCommand** y **SqlConnection** respectivamente.
4. Los atributos se deben inicializar en el constructor de la clase.
5. El método ***GuardarRegistro()***, va a insertar un nuevo registro en la base de datos con los datos del payaso y el nombre del alumno, la fecha debe guardarse en un tipo string con el siguiente formato **“dd/mm/yyyy”**

**IGuardar<T>**

1. El método Guardar() va a recibir como único parámetro el ítem a guardar

**Serializadora**

1. Implementar *IGuardar<T>*, se va a encargar de guardar un objeto de tipo ***Payaso*** *en el escritorio en formato Json.* 
   * 1. Implementar **DatosException** donde sea necesario.
     2. Validar si el archivo fue creado.

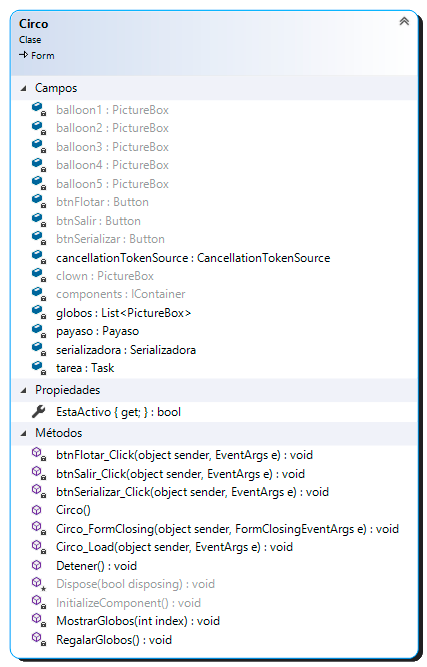
**Logger**

1. El método estático ***Log()*** va a recibir el mensaje de una excepción por parámetro y va a escribir en un archivo de texto en el escritorio el mensaje junto con la fecha y la hora en ese momento.
   * 1. Implementar **DatosException** donde sea necesario.

**Test Unitarios**

1. Crear un test unitario que pruebe la serialización en formato ***JSON.***
2. Crear un test unitario que pruebe la excepción **DatosException.**
3. Crear un test unitario que pruebe el método de extensión**.**

**Formulario**

****

1. El formulario tendrá 5 atributos:
   * 1. ***globos*** de tipo ***List<PictureBox>***.
     2. ***payaso*** de tipo ***Payaso***.
     3. ***serializadora*** de tipo ***Serializadora***.
     4. **cancellationTokenSource** de tipo **CancellationTokenSource**.
     5. **tarea** de tipo **Task**.
2. La propiedad EstaActivo de solo lectura va a estar encargada de validar si la tarea actual no es nula y si no está en un estado activo
3. El constructor deberá instanciar los atributos ***globos***, ***serializadora*** y ***payaso***.
4. El manejador del evento ***Load*** del formulario deberá:
   * 1. Asignar como manejador del evento ***AparecerGlobo*** de ***Payaso*** al método del formulario ***MostrarGlobos***.
5. El botón flotar es el encargado de llamar al método ***RegalarGlobos.***
6. El método ***RegalarGlobos*** va a validar que la tarea no se haya iniciado aún o haya sido completada, va a instanciar el objeto de tipo **cancellationTokenSource** yuna vez validado eso se va a encargar de de asociar el método ***Flotar*** de payaso a la tarea e iniciarlo si no se encuentra activo.
7. El método Detener va a cortar el hilo si se encuentra activo.
8. Agregar el método ***MostrarGlobos*** que va a recibir el índice del globo.
   * 1. La modificación de los ***globos*** se deberá realizar desde el hilo principal (en el que se crearon los controles).
9. El botón serializar va a llamar al método guardar de la clase serializadora para guardar un payaso.
   * 1. Informar si el archivo se pudo crear o no.
10. El botón salir va a cerrar el formulario, deberá preguntarle al usuario si está seguro que desea salir(implementar evento form\_closing) y además va a controlar cerrar los hilos .

Al finalizar, colocar la carpeta de la Solución completa en un archivo ZIP que deberá tener como nombre Apellido.Nombre.division.zip y compartir este por Slack sólo con el docente titular de la cursada.